LOVELY PROFESSIONAL UNIVERSITY

Transforming Education Transforming India

Artificial Intelligence – INT404

Best Route Finding

RAJ KUMAR	RAHUL PANDEY	PRASHANT GIRI
11803481	11803255	11803253
B39	B38	B37

Jasleen Kaur- Assistance Professor

Lovely Professional University

Jalandhar, Punjab

Student Declaration

We have to declare that this report is completely written by us.Nothing of this report is copied from any other sources.Each and every information inserted from other sources have been duly appreciated.We aver that if any part of this report is found to be mock up,we shall be completely responsible for this.

11803255 - Rahul Pandey RK18MSB38

11803481 – Raj Kumar RK18MSB39

11803253 - Prashant Giri RK18MSB 37

Place: -Jalandhar

Date: -Apr 06, 2020

BONAFIDE CERTIFICATE

This is to be certified that this "**Rahul Pandey**, **Raj kumar, Prashant Giri**" student of B. Tech (CSE) of Lovely Professional University has worked under my supervision and guidance for this project work and prepared a project Report with the title "**Best Route Finding**".

The project Report which these are submitting is his genuine and original work to Best of my knowledge.

Jasleen Kaur

Assistant

Professor 25340

Department of Intelligent Systems

Tables of contents

Title **Page no.**

Introduction

Best route finding is basically based on Dijkstra's algorithm. Dijkstra's algorithm is an algorithm we can use to find shortest distances or minimum costs depending on what is represented in a graph. You're basically working backwards from the end to the beginning, finding the shortest leg each time. Suppose we must travel between the Patna to Delhi, From Patna there are two routes came under knowledge one is via Gorakhpur and other is via Varanasi. Patna via Gorakhpur route is 880KM and another route that is via Varanasi is 930KM.So, this Intelligence help the people that which route are best route by KM and analyzing about any other reason. And process will be done by Dijkstra's algorithm. In short we can say that the process by which we find Best and shortest route that is process is called Dijkstra's algorithm.

Dijkstra’s Algorithm

Dijkstra’s algorithm helps us to find the best route or shortest route between any two point. It is different from minimum spanning tree because distance between two vertices might not include all the vertices of graph.

How Dijkstra’s Algorithm Works

Dijkstra’s algorithm Works on the basis that any sub-route B->D of the shortest route A->D between vertices A and D is also the shortest route between B and D.

Dijkstra used this property in the opposite direction i.e. we are overestimating the distance of each vertex from starting vertex. Then we visit each node its neighbors to find the shortest sub-route to those neighbors.

The Algorithm used to greedy approach in the sense that we find the next best solution hoping that the result is the best solution for whole problem.

Scope of Project

Now a day this Project is very useful. Because people are roaming here and there in all over the World. But they don't know about the shortest path or route of destination. So, this intelligence are very helpful to reached the destination.

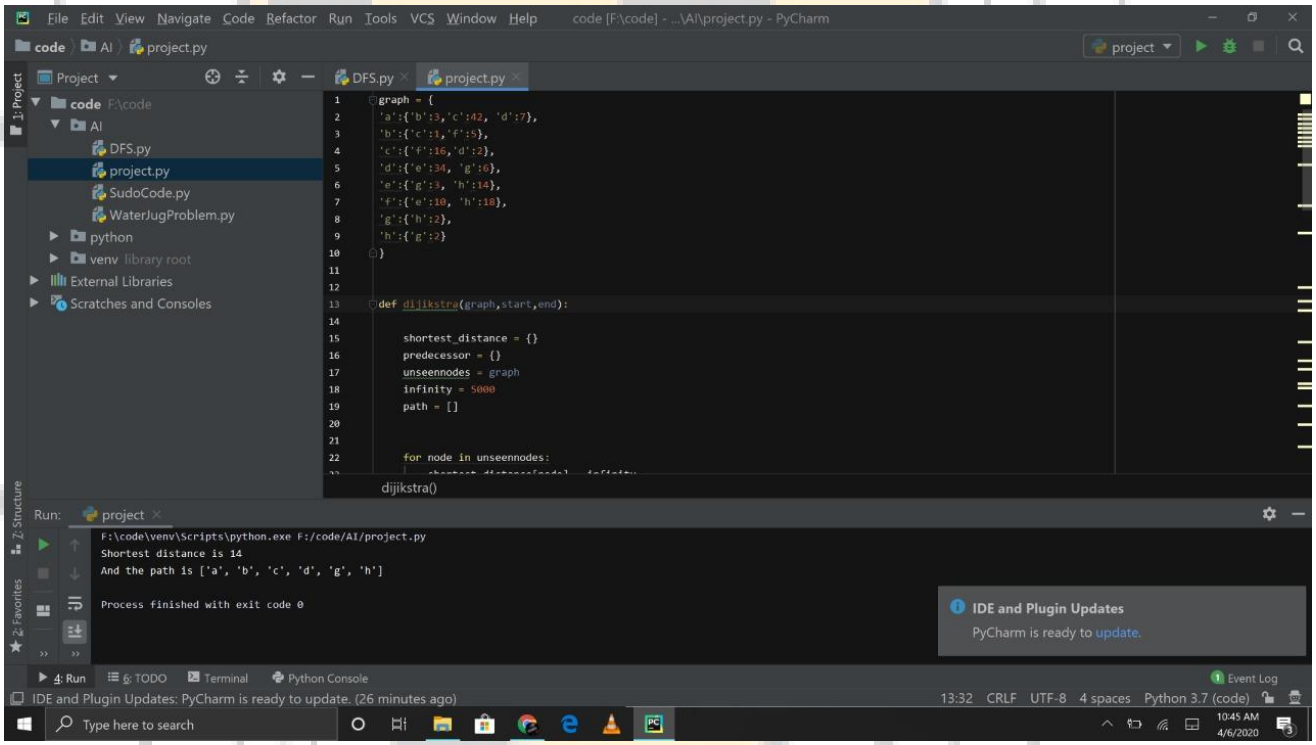
Programs

```
graph = {
'a': {'b':3,'c':42, 'd':7},
'b': {'c':1,'f':5},
'c': {'f':16,'d':2},
'd': {'e':34, 'g':6},
'e': {'g':3, 'h':14},
'f': {'e':10, 'h':18},
'g': {'h':2},
'h': {'g':2}
}


def dijikstra (graph, start, end):

    shortest_distance = {}
    predecessor = {}
    unseennodes = graph
    infinity = 5000
    path = []


    for node in unseennodes:
        shortest_distance[node] = infinity

    shortest_distance[start] = 0


    while (unseennodes):
        min_distance_node = None
```

```
 for node in unseennodes:
            if min_distance_node is None:
                min_distance_node = node


elif shortest_distance[node] <
shortest_distance[min_distance_node]:
                min_distance_node = node


           path_options = graph[min_distance_node].items()

        for child_node, weight in path_options:

            if (weight + shortest_distance[min_distance_node] <
shortest_distance[child_node]):

                shortest_distance[child_node] = weight +
shortest_distance[min_distance_node]

                predecessor[child_node] = min_distance_node


        unseennodes.pop(min_distance_node)


    currentNode = end

    while (currentNode != start):

        try:
            path.insert(0,currentNode)
            currentNode = predecessor[currentNode]
        except KeyError:
            print('Path not reachable')
```

```
        break
        path.insert(0,start)


        if (shortest_distance[end] != infinity):
            print('Shortest distance is ' + str(shortest_distance[end]))
            print('And the path is ' + str(path))

dijikstra(graph, 'a', 'h')
```

Output



Advantages

- It is used in Google Maps
- It is used in finding Shortest Path.
- It is used in geographical Maps
- To find locations of Map which refers to vertices of graph.
- Distance between the location refers to edges.
- It is used in IP routing to find Open shortest Path First.
- It is used in the telephone network.

Disadvantages

- It does a blind search, so wastes a lot of time while processing.
- It can't handle negative edges.
- It leads to the acyclic graph and most often cannot obtain the right shortest path.
- We need to keep track of vertices that have been visited.
-